基于国密算法 SM2、SM3、SM4 的高速混合加密系统硬件设计 *

李建立，莫燕南，粟　涛†，陈弟虎

(中山大学 电子与信息工程学院，广州 510006)

摘　要：随着电子信息技术的快速发展，数据的安全性问题日益严峻，传统单一制密码算法在安全性与运算速率上已不能满足要求。为了解决大数据时代下所提出的加密需求，提出了一种基于国密算法 SM2、SM3、SM4 的高速混合加密系统的硬件设计方案，并针对 SM2 和 SM4 算法的底层硬件结构进行优化：对 SM2 算法采用 Karatsuba-Ofman 模乘器与点运算并行方案进行优化；对 SM4 算法提出了一种基于复合域 S 盒的二次流水全展开硬件架构。实验结果表明，该系统所实现的各算法电路均具有较高性能：SM2 的点乘时间缩短至 68.37 μs；SM3 的杂凑值生成仅需 0.71 μs；SM4 吞吐率最高达 53.76 Gbit/s。相比同等安全性的 SM2 算法，该混合系统在加密时间上减小了 26.67%，具有实用价值。与相关工作进行对比分析，可证明该方案在安全性和加解密性能上均有一定优势。

关键词：SM2 公钥密码算法；SM4 分组加密算法；硬件设计；混合加密

中图分类号：TP391　　doi: 10.19734/j.issn.1001-3695.2022.02.0072

Hardware design of high-speed hybrid encryption system based on SM2, SM3 and SM4 algorithm

Li Jianli, Mo Yannan, Su Tao†, Chen Dihu

(*School of Electronics & Information Technology*, *Sun Yat-Sen University*, *Guangzhou* 510006, *China*)

Abstract: With the rapid development of electronic information technology, the security of data has become increasingly serious. The traditional single cipher algorithm can not meet the requirements any more. In order to solve these problems, this paper proposed a hardware design scheme of high-speed hybrid encryption system based on SM2, SM3, SM4, in which the hardware structure of SM2 and SM4 algorithms is optimized. Besides, this paper optimized the SM2 algorithm with Karatsuba-Ofman modular multiplier and parallel with point operation. For SM4 algorithm, this paper designed a subpipelining hardware architecture based on composite field S-box so as to improve its throughout. The experimental results showed that all the algorithm circuits implemented by the system have high performance: the point multiplication time of SM2 was 68.37 μs; the hash generation of SM3 only took 0.71 μs; the throughput of SM4 was up to 53.76 Gbit/s. Compared with the SM2 algorithm with the same security, this hybrid system had 26.67% less encryption time. By comparing and analyzing with related work, it can be proved that this scheme has certain advantages in security and encryption and decryption performance.

Key words: SM2 algorithm; SM4 algorithm; hardware design; hybrid encryption

0　引言

近年来，随着计算机技术的快速发展以及大数据、云计算、物联网等技术产业的横空出世，人们在生活与生产中越来越离不开各种信息数据的传递与交互。信息技术在推动经济与社会发展的同时，也面临着巨大的安全隐患：网络攻击、信息泄露、隐私数据被盗等事件频发，信息安全问题日益加剧。另外，各类侧信道密码攻击技术的发展也使得各种单一加密算法面临着严峻的安全性挑战。与此同时，5G 技术的日益成熟让人们步入了高带宽时代，也对数据的加密速度提出了更高的要求。在即时通信、车联网应用、远程医疗等对延时要求较高的场景下，加密系统的数据处理速度变得十分关键。因此，在大数据时代下，研究如何在高速传输的情况下保障数据信息安全至关重要。

混合加密技术是结合了多种加密体制和算法的一种新型加密技术，可以根据实际需求选择不同的加密体制进行加密方案的构建。目前，许多国内外学者对混合加密体制进行了大量的研究与分析。Mostafaa 等人[1]提出了一种基于 AES 和 ECC 的轻量级混合加密硬件系统，并通过优化 AES 算法使得整体方案更加适合硬件资源有限的场景。V. Shende 等人[2]则结合了 AES 和 RSA 两种算法的优点提出了一种混合加密方案，提高了系统的安全性。在国密算法方面，文献[3, 4]分别选择了 SM4-ECC 和 SM4-SM2 的算法组合构建其加密方案，并从软件层面证明所提出的混合加密体制在安全性与扩散性系数上均高于单一制算法。文献[5]则通过研究 SM2、SM3 与 SM4 算法在变电站局域网中的应用情况，设计出一种基于国密算法的云存储混合加密方案。X. Zheng 等人[6]利用软硬件协同设计的方法，提出了同样基于 SM2、SM3 和 SM4 的加解密和数字签名方案，与相关工作相比其处理速度提高了 10%以上。总体而言，混合加密体制不仅提高了算法的安全性，同时也能兼顾加解密的高效性与密钥管理的便利性，优于传统单一制加密算法。

国密算法 SM2 和 SM4 是目前国内较为主流国产密码算法，在安全性研究与硬件实现上已有大量的研究成果。文献[7]基于 Montgomery 模乘算法，设计了一种高速双域 SM2 模乘器，并提出了一种新型的 Wallace 树乘法单元，节省了大量资源。文献[8]提出了一种基于改进蒙哥马利点乘算法的并行点运算架构，通过坐标压缩以减少模乘步骤。文献[9]针对

收稿日期：2022-02-14；修回日期：2022-04-20　**基金项目：**广东省重大科技计划资助项目(2021B110127007)

作者简介：李建立(1997-)，男，广东广州，硕士研究生，主要研究方向为加密算法、硬件加速；莫燕南(1997-)，男，广东深圳，硕士研究生，主要研究方向为 AI 加速、人工智能；粟涛(1977-)，男(通信作者)，湖南益阳，副教授，博士，主要研究方向为人工智能，芯片与应用系统设计(sutao@mail.sysu.edu.cn)；陈弟虎(1963-)，男，四川绵阳，教授，博士，主要研究方向为集成电路设计方法，深度学习与图像识别技术.

低端 FPGA 芯片，从算法和硬件实现两方面优化了 SM2 的模乘和模逆运算，提高了硬件资源的复用率，实现了面积和速度的平衡。而在 SM4 算法硬件架构的研究上，Gao[10]和 Wang H[11]均采用了 32 轮轮间流水全展开型架构，降低了每一组数据加解密的平均时间，提供了极高的吞吐率。文献[12]提出了一种轮密钥扩展与加解密模块硬件复用的架构，其占用的资源仅为传统设计的 55%。文献[13]和 Fu H 等人[14]则在轮间流水全展开的基础上对轮函数内部进行了二次流水切分，减少了电路的关键路径。

从文献分析与研究结果看，目前在混合加密方面国外的研究重点偏重于 AES、RSA 以及 ECC 这类国际算法，而国内对国密算法的混合加密研究关注点主要集中在算法安全性方面，对混合加密算法的硬件优化和性能提升上的研究比较欠缺。此外，在国密算法 SM2 与 SM4 的硬件实现研究上也仍存在一定的优化提升空间。

基于上述现状问题的分析，本文结合三种国密算法各自的优势，提出了一种基于国密算法 SM2、SM3、SM4 的混合加密硬件系统设计，在提高了加密系统安全性的同时可完成大量信息数据的高速加解密运算。同时，还针对 SM2 与 SM4 算法的结构对其硬件架构进行了优化设计，最终设计出一个集运算速率、硬件资源利用率以及数据安全性于一体的高速混合加密硬件系统，以解决信息化建设中对应用数据提出的高速率、高吞吐率加解密传输需求。

1　算法原理与混合加密系统设计原理

1.1　SM2 算法原理

SM2 椭圆曲线密码算法是我国公钥密码算法标准，于 2010 年 12 月首次公开。相关研究表明，SM2 算法的安全性达到了公钥密码算法的最高安全级别[15]。

1.1.1 SM2 曲线参数选择

SM2 算法具有与 ECC 算法相同的数学概念基础，是一种基于求解椭圆曲线离散对数问题的加密算法。椭圆曲线是一种特殊的曲线，它实际上是 Weierstrass 方程所构成的平面光滑曲线，定义在伪梅森素数域 GF(p)上的 SM2 椭圆曲线表达式为

$$E: y^2 = x^3 + ax + b \tag{1}$$

其中，$a, b \in$ GF(p)，且 $4a^3+27b^2 \neq 0 \bmod p$。在国家密码标准中，推荐的定义在 256 位素数域上的 SM2 椭圆曲线系统参数为(所有值均以 16 进制表示)：

素数 p = FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 FFFFFFFF FFFFFFFF

系数 a = FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 FFFFFFFF FFFFFFFC

系数 b = 28E9FA9E 9D9F5E34 4D5A9E4B CF6509A7 F39789F5 15AB8F92 DDBCBD41 4D940E93

1.1.2 SM2 算法层次结构

SM2 算法在结构上具有明显的层次性特点，其自顶向下可以划分为四个层次，如图 1 所示。顶层为 SM2 应用层，包括数字签名、密钥交换和公钥加密，主要通过调用点乘计算模块来实现不同的加密功能。群运算层为点乘模块，是 SM2 的核心运算，通过调用点加与倍点实现。点运算层分为点加和倍点两个模块，分别实现椭圆曲线上两个基础计算功能。底层则是最基本的模运算单元，包括模加、模减、模乘以及模逆运算，分别实现有限域上的各种模运算。本文对于 SM2 算法的硬件加速研究主要集中在点运算层与模运算层，通过并行化点运算以及优化模乘运算实现高效运算，从而提高 SM2 整体性能。

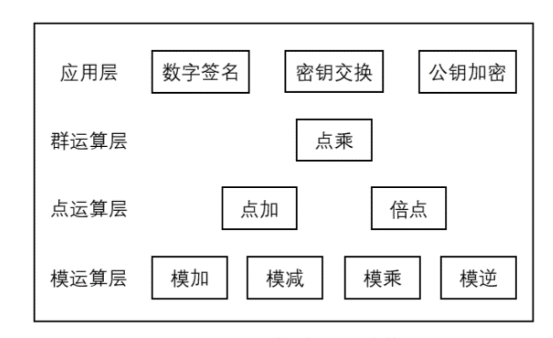


图 1　SM2 算法层次结构

Fig. 1　Hierarchy of SM2

1.2　SM3 算法原理

SM3 算法是一种分组迭代的杂凑算法，可以将任意长度的输入转为恒为 256bit 的摘要值输出，由国家密码管理局于 2010 年 12 月 17 日发布。SM3 算法主要分为填充分组以及迭代压缩两个步骤，其计算流程如图 2 所示。

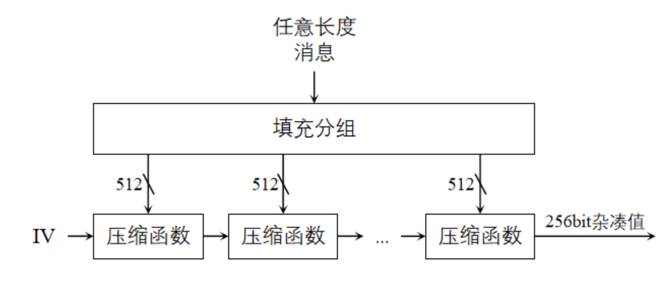


图 2　SM3 算法计算流程

Fig. 2　Calculation process of SM3

填充分组：假设输入的长度为 l 比特。首先将比特“1”添加到消息的末尾，再添加 k 个“0”，k 为满足 $l+1+k \equiv 448 \bmod 512$ 的最小非负整数。然后再添加一个长度 l 二进制表示的比特串。填充后的比特长度为 512 的倍数，并按 512bit 一组分组。

迭代压缩：对分组后的消息按下列方式进行迭代计算：

$$V^{(i+1)} = CF(V^{(i)}, B^{(i)}), i = 0,1,2,\ldots,n-1 \tag{2}$$

其中，$n=(l+k+65)/512$；CF 是压缩函数，$V(0)$为 256bit 初始值 IV，$B^{(i)}$是消息分组，$V^{(n)}$是最后的杂凑值结果。

1.3　SM4 算法原理

SM4 分组密码算法是我国密码管理局于 2006 年公布的国内第一个商用密码算法，对于我国密码学研究的发展起了极大的推动作用[16]。SM4 算法是一种分组算法，分组长度和密钥长度均为 128bit。SM4 内部的加解密运算和密钥扩展运算都是 32 轮的非线性迭代结构，而非线性变换里的基本运算单元为 S 盒。其主要运算结构以字(Z_2^{32})为单位，一次运算为一轮变换。定义 ⊕ 为 32bit 异或运算，$<<<i$ 为 32bit 循环左移 i 位；加密密钥以 MK 表示；轮密钥以 rk_i 表示，系统参数以 $FK=(FK_0,FK_1,FK_2,FK_3)$ 表示，固定参数以 $(CK_0,CK_1,\ldots,CK_{31})$表示。

1.3.1 加解密算法

加密算法过程如算法 1 所示。

算法 1　SM4 加密算法

输入：$(X_0,X_1,X_2,X_3) \in (Z_2^{32})^4$，$rk_0,rk_1,\ldots,rk_{31} \in Z_2^{32}$ 为轮密钥。

输出：$(Y_0,Y_1,Y_2,Y_3) \in (Z_2^{32})^4$。

1. i=0,1,2,...,31:

$X_{i+4} = F(X_i,X_{i+1},X_{i+2},X_{i+3},rk_i) = X_i \oplus T(X_i \oplus X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus rk_i)$

2. $(Y_0,Y_1,Y_2,Y_3) = (X_{35},X_{34},X_{33},X_{32})$

其中，F 为轮变换函数；T 为合成置换，包含了非线性变换 τ 和线性变换 L，即 $T(.)=L(\tau(.))$；τ 内部为四个并行 S 盒，L 为线性变换。合成置换 T 过程如算法 2 所示。

算法 2　合成置换 T 算法

输入：$A=(a_0,a_1,a_2,a_3)\in(Z_2^{8})^4$。

输出：$C=(c_0,c_1,c_2,c_3)\in(Z_2^{8})^4$。

1．$B=(b_0,b_1,b_2,b_3)=\tau(A)=(Sbox(a_0),Sbox(a_1),Sbox(a_2),Sbox(a_3))$

2．$C=L(B)=B\oplus(B<<<2)\oplus(B<<<10)\oplus(B<<<18)\oplus(B<<<24)$

解密算法与加密算法的运算流程相同，只是轮密钥的使用次序与加密相反。

1.3.2 密钥扩展算法

密钥扩展运算过程如算法 3 所示。

算法 3　密钥扩展算法

输入：$MK=(MK_0,MK_1,MK_2,MK_3), MK_i\in Z_2^{32}(i=0,1,2,3)$；

输出：$rk_i\in Z_2^{32}(i=0,1,2,\ldots,31)$

1．$(K_0,K_1,K_2,K_3)=(FK_0\oplus MK_0,FK_1\oplus MK_1,FK_2\oplus MK_2,\ FK_3\oplus MK_3)$

2．i=0,1,2,...,31:

$rk_i=K_{i+4}=K_i\oplus T'(K_{i+1}\oplus K_{i+2}\oplus K_{i+3}\oplus CK_i)$

其中，T'中的线性变换为 L'：$(B<<<13)\oplus(B<<<23)$，其余部分与加解密运算中的 T 变换相同。

1.4　基于 SM2、SM3、SM4 算法的高速混合加密系统

SM2 非对称算法得益于椭圆曲线密码体制，其公钥可对外公开，可实现密钥的便捷管理，信息传输的安全性也较高。但是由于涉及大位宽的复杂模运算，SM2 算法的加解密速度普遍较慢，仅适用于加密小块数据。SM3 算法属于单向加密的密码体制，对于明文信息只能计算出杂凑值，而无法从杂凑值反向推算出对应明文数据。SM4 对称算法虽然结构简单、复杂度低、加解密速度快，但是其需要额外的安全信道分发密钥，密钥管理复杂且安全性难以保证。

本文在研究了 SM2、SM3 以及 SM4 三种算法各自的特点后，提出了一种兼顾加密安全性、高速性以及密钥管理便利性的混合加密系统，其基本原理如下：基于大数据时代高速率加解密传输的需求，采用经本文硬件优化后的 SM4 算法快速加解密海量的应用数据；利用 SM2 算法安全性高、密钥管理简单的特点，采用同样经优化后的 SM2 算法对 SM4 密钥进行加解密；另外，本系统还采用 SM3 杂凑算法对加解密前后的明文进行杂凑值对比验证，防止信息在传输过程中被窜改。由于混合使用了三种不同体制的加密算法，本文提出的混合加密系统在传输密钥时不再需要额外的安全信道，同时还可以实现大量数据的高速加解密与验证运算，其加解密流程如图 3 所示。

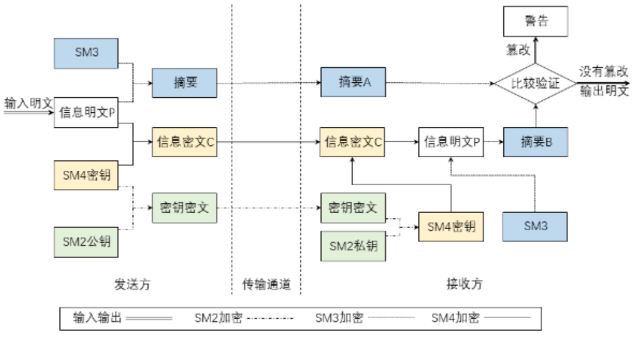


图 3　混合加密系统加解密流程

Fig. 3　Encryption and decryption process of hybrid encryption system

假设需要加密传输的数据为信息明文 P，且发送方和接收方之间的传输通道是不安全的，则具体的加解密传输过程如下：

a) 首先接收方使用 SM2 算法生成配对的公钥和私钥，并在加密计算前先将公钥发送给发送方；

b) 发送方接收到 SM2 公钥后，使用 SM2 算法加密 SM4 密钥生成密钥密文，同时使用 SM4 算法加密信息明文 P 生成信息密文 C，另外再使用 SM3 算法计算 P 中首尾两组数据块的摘要值。待三种算法均计算完成后，将三组数据打包，通过传输通道一同发送给接收方；

c) 接收方在收到数据后，首先利用 SM2 私钥，通过 SM2 算法将 SM4 密钥解密出来。然后利用 SM4 密钥，通过 SM4 算法解密所收到的信息密文 C，得到尚未验证的信息明文 P。接着使用 SM3 算法计算得到信息明文 P 的摘要值 B，并与所收到的摘要值 A 进行对比验证：若一致，则代表数据在传输过程中没有被窜改，可输出所解密的信息明文 P，完成解密；若两组摘要值不一致，则说明数据在传输中已被窜改，解密出来的数据存在安全风险，系统发出警告并退出解密。

从理论上分析本文提出的混合加密系统可以发现：在加密过程中最耗时的步骤为海量数据明文的 SM4 加密计算；而因为解密过程是三种算法串行执行的，所以决定解密速度的步骤主要为对密钥密文的 SM2 解密计算，以及对大量信息密文的 SM4 解密计算。基于以上分析可知，若想提高混合加密系统的整体性能，需考虑提升 SM2 和 SM4 算法的计算速率和硬件效率。因此，本文的研究重点将集中在对 SM2 和 SM4 算法的硬件优化和性能提升上，采用并行化与流水线技术对两个算法的硬件架构进行优化设计，解决制约加解密速度提升的瓶颈，保证混合加密系统能够满足高性能密码运算的需求。

2　SM2 算法的硬件加速设计

椭圆曲线的点乘运算是 SM2 算法的核心运算，其运行速度决定了 SM2 算法的速度。点乘运算一般涉及大量大数运算，而这些运算操作用软件实现会非常复杂且效率低下，因此通常采用硬件优化的实现方式以提高 SM2 算法的性能。本文在综合考虑性能与资源的情况下，采用自底向上的思路，对 SM2 算法的硬件实现进行优化设计。首先从底层的模乘运算出发，对模乘器的硬件实现采用 Karatsuba-Ofman 算法进行优化设计，降低计算复杂度与硬件资源的消耗。其次，本文还分析研究了不同架构下点加和倍点运算的硬件性能提升与资源消耗情况，在资源合理运用的情况下实现了 SM2 算法的快速点乘运算。

2.1　模乘运算优化设计

SM2 算法在进行点乘运算的过程中会频繁地调用乘法器进行大数模乘运算，故实现模乘的快速运算是提高 SM2 算法性能的一个关键。常见的模乘方法主要有 Barrett 算法、Montgomery 算法、大数乘法搭配快速模约减等[17]。Barrett 算法除了乘法以外还采用了除法操作，对于硬件实现来说该算法在运算时间与资源消耗上都非常低效。而 Montgomery 算法虽然通过移位和加法操作代替了除法操作，但是其运算结果并非模乘的最后结果，仍需要进一步的转换运算，更适合用于多次乘法的模幂运算，对于单次模乘计算并无优势。因此，本文选择采用大数乘法搭配快速模约减的方式实现模乘运算。

在大数乘法方面，运用 Karatsuba-Ofman 算法的分治思想，将 256 位的乘法拆分成三次 128 位乘法，并以基于 129 位乘法器的方式完成大数乘法运算。相比于直接使用 256 位乘法器[18]，Karatsuba-Ofman 算法虽然需要更多的时钟周期来完成，但是减少了 25%的计算量，硬件资源消耗更小，提高了运算效率与资源利用率，基于该算法实现的 256 位乘法如算法 4 所示。

算法 4　256 位 Karatsuba-Ofman 算法

输入：256 位整数 A，B，且有 $A=a_H\times2^{128}+a_L$，$B=b_H\times2^{128}+b_L$。

输出：$C=A\times B$，C 为 512 位整数。

1．$P_L=a_L\times b_L$，$a_S=a_H+a_L$，$b_S=b_H+b_L$

2．$P_H=a_H\times b_H$，$C_1=P_H+P_L$，$C_2=\{P_H,P_L\}-C_1\times2^{128}$

3．3．$P_S=a_S\times b_S$，$C=C_2+P_S\times2^{128}$

该算法中包含了三次 128 位的加/减法、两次 512 位的加/减法、两次 128 位的乘法以及一次 129 位的乘法。同时，算法中 2 的幂次运算在硬件中可以通过移位操作实现，以节省硬件消耗与计算时间。根据算法 4，提出如图 4 所示的 256 位 Karatsuba-Ofman 乘法硬件设计：

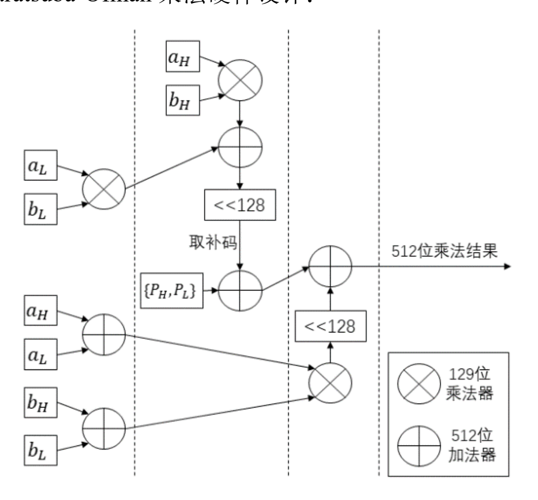


图 4　256 位 Karatsuba-Ofman 乘法硬件设计

Fig. 4　Hardware design for Karatsuba-Ofman multiplication in 256 bits

该硬件设计采用一个 129 位的乘法器、两个并行的 512 位加法器以及移位寄存器实现，同时对加法运算进行二级并行处理，将 256 位的乘法运算降低至三个时钟周期。其中乘法器负责 128 位与 129 位的乘法运算，加法器负责 128 位与 512 位的加减法运算，通过是否取补码实现。该设计在缩短运算周期的同时也实现了较低的硬件开销，资源利用率达到了 88.8%。

在取模方面，由于本文选取的素数 p 为伪梅森素数。根据其性质，该素数可表示为少量的 2 的幂的和或差，本文选取的素数则可改写为 $p_{256}=2^{256}-2^{224}-2^{96}+2^{64}-1$。根据该表达式，可以采用快速模约减算法快速地对 512 位的乘法结果取模。参考文献[19]中的快速模约减算法，完成整体模乘器的硬件电路设计如图 5 所示。快速模约减算法通过简单的截位拼接与加减法即可完成取模运算，省去了常规取模中繁琐的乘法与除法运算，有利于缩短整体的运算时间、节省硬件的资源消耗。

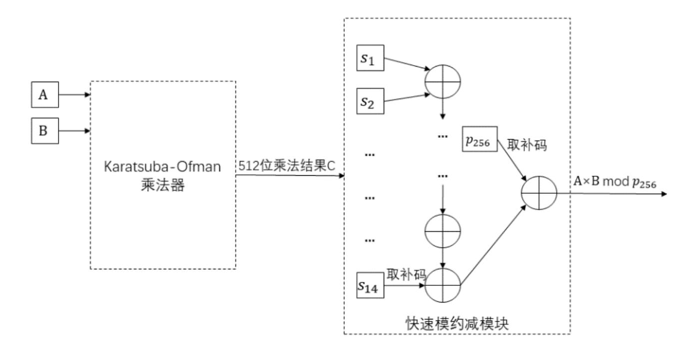


图 5　模乘器硬件结构

Fig. 5　Hardware structure of modular multiplication

该模乘器采用了 Karatsuba-Ofman 算法与快速模约减算法相匹配的方式，仅利用低位宽加法、乘法以及移位等简单的操作，最终在四个周期内完成了复杂的 256 位模乘运算。

2.2　点运算并行优化设计

点运算层是实现点乘运算的基础，因此对点加和倍点运算进行优化加速也非常重要。在 Jacobian 投影-仿射混合坐标下实现点加、在 Jacobian 投影坐标下实现倍点的组合方式可以避免耗时的模逆运算，同时该组合方式也是众多坐标系中运算复杂度最低的选择。Jacobian 投影-仿射混合坐标下点加运算与 Jacobian 投影坐标下倍点运算的定义分别如算法 6、7 所示[19]：

算法 6　Jacobian 投影-仿射混合坐标下点加运算

输入：$P=(X_1,Y_1,Z_1)$，$Q=(X_2,Y_2,1)$，$P\neq Q$，$Z_1\neq 0$。

输出：$P+Q=(X_3,Y_3,Z_3)$。

$X_3=(Y_2Z_1^3-Y_1)^2-(X_2Z_1^2-X_1)^2(X_2Z_1^2+X_1)$

$Y_3=(Y_2Z_1^3-Y_1)(X_1(X_2Z_1^2-X_1)^2-X_3)-Y_1(X_2Z_1^2-X_1)^3$

$Z_3=(X_2Z_1^2-X_1)Z_1$

算法 7　Jacobian 投影坐标下倍点运算

输入：$P=(X_1,Y_1,Z_1)$，$Z_1\neq 0$，$a=p-3$。

输出：$2P=(X_3,Y_3,Z_3)$。

$X_3=[3(X_1+Z_1^2)(X_1-Z_1^2)]^2-8X_1Y_1^2$

$Y_3=3(X_1+Z_1^2)(X_1-Z_1^2)(4X_1Y_1^2-X_3)-8Y_1^4$

$Z_3=2Y_1Z_1$

对于点加算法，通过分析算法中各步骤数据间的依赖关系，将不存在相关性的运算进行并行加速，从算法实现层面降低点加运算时间。对算法步骤进行拆分，根据运算相关性提出了点加计算的二级并行加速方案，如算法 8 所示。

算法 8　Jacobian 投影-仿射混合坐标下二级并行点加运算

输入：$P=(X_1,Y_1,Z_1)$，$Q=(X_2,Y_2,1)$，$P\neq Q$，$Z_1\neq 0$。

输出：$P+Q=(X_3,Y_3,Z_3)$。

模乘器 0　模乘器 1　模加减器 0　模加减器 1

1　$A=Z_1\times Z_1$　$B=Y_2\times Z_1$　NA　NA

2　$A\times B$　$C=A\times X_2$　$A=AB-Y_1$　$B=C-X_1$

3　$Z_3=B\times Z_1$　$D=B\times B$　$E=C+X_1$　NA

4　$A\times A$　$D\times E$　$X_3=AA-DE$　NA

5　$D\times X_1$　$B=D\times B$　$C=DX_1-X_3$　NA

6　$A\times C$　$B\times Y_1$　$Y_3=AC-BY_1$　NA

由算法 6 可知，点加运算需要进行 9 次的模乘计算与 3 次的模平方运算，若采用传统的循环串行计算方式则至少需要十二轮运算来完成。而本文提出的二级并行方案则通过采用两个模乘器的方法进行加速，对乘法运算采用并行的计算方式处理，从而将运算迭代缩短至六轮。根据所提出的点加二级并行方案，点加模块在硬件实现上需要实例化两个模乘器、三个模加减器，并通过共享的寄存器(A、B、C、D、E)存储与调用中间的计算结果。在研究了串行、二级并行以及三级并行[20]三种方案每轮迭代中最长的运算路径后，可得出如表 1 所示的运算时间对比表：

表 1　不同并行度点加方案的运算时间对比

Tab. 1　Comparison of operation time for point addition schemes with different parallelism

并行度	运算周期	加速比
串行	$11M\times 5cycles+2A/S\times 1cycle+1cycle=58cycles$	NA
二级并行	$6M\times 5cycles+4A/S\times 1cycle+1cycle=35cycles$	1.657
三级并行	$5M\times 5cycles+5A/S\times 1cycle+1cycle=31cycles$	1.871

其中 M 代表模乘，A/S 代表模加与模减运算。由于在模乘运算以及每轮迭代后，共享寄存器需要额外一个时钟周期进行数据的暂存与传输，因此在点运算中模乘实际需要 5 个时钟周期。

通过对比可知，二级并行方案和文献[20]中的三级并行方案与传统串行方案相比在速度上均有极大的提升。而三级并行方案[20]虽然在运算时间上是三种方案中最快的，但是该方案需要例化三套模乘器与模加减器，其在硬件消耗上也是三者中最高。因此，本文在综合考虑性能与硬件资源的因素后，最终决定采用二级并行的方式实现点加运算。

同理，对于倍点算法本文采用相同的分析方式提出了如算法 9 所示的二级并行加速方案。

算法 9　Jacobian 投影坐标下二级并行倍点运算

输入：$P=(X_1,Y_1,Z_1)$，$Z_1\neq 0$，$a=p-3$；

输出：$2P=(X_3,Y_3,Z_3)$

	模乘器 0	模乘器 1	模加减器 0	模加减器 1	模加减器 2
1	$A=Y_1\times Z_1$	$Z_1\times Z_1$	$B=Y_1+Y_1$	$C=X_1+Z_1^2$	$C=X_1-Z_1^2$
2	$A=C\times D$	$B=B\times B$	$Z_3=A+A$	$C=B+B$	NA
3	$C=C\times X_1$	$D=B\times X_1$	$A=A+A$	$A=A+2A$	NA
4	$A\times A$	$B\times Y_1$	$B=BY_1+BY_1$	$X_3=AA-C$	$C=D-X_3$
5	$A\times C$	$B\times Y_1$	$Y_3=AC-BY_1$	NA	NA

倍点运算需要进行 6 次的模乘计算与 4 次的模平方运算，根据本文提出的二级并行方案，可将倍点运算的迭代减至五轮。在硬件实现上，倍点模块同样需要实例化两个模乘器与三个模加减器，以及四个共享寄存器。倍点运算三种方案的运算时间对比如表 2 所示。

表 2　不同并行度倍点方案的运算时间对比

Tab. 2　Comparison of operation time for double point schemes with different parallelism

并行度	运算周期	加速比
串行	$9M\times 5cycles+1A/S\times 1cycle+1cycle=47cycles$	NA
二级并行	$5M\times 5cycles+5A/S\times 1cycle+1cycle=31cycles$	1.516
三级并行	$4M\times 5cycles+6A/S\times 1cycle+1cycle=27cycles$	1.741

可以看到，二级并行方案在运算周期上比串行方案有一定的提升。为了提高硬件利用率，本文在倍点运算的实现同样选择了兼顾性能与资源消耗的二级并行方案。

2.3　点乘运算的实现

点乘运算的实质是对椭圆曲线坐标进行多次点加与倍点计算的结果，故在电路实现上需要频繁调度点加与倍点模块。由于本文采用不同的坐标系实现点加和倍点运算以减少耗时的模逆运算次数，因此还需要相应的坐标转换模块，从而将最终的点乘结果转换至仿射坐标系下。图 6 为点乘运算的组成模块。

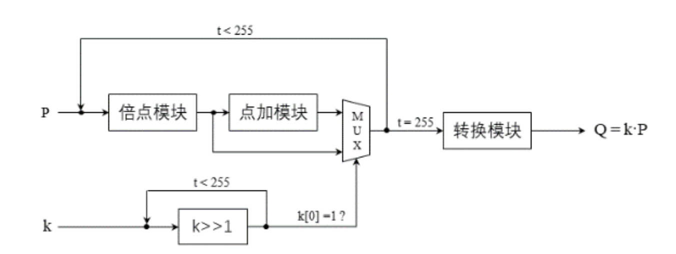


图 6　点乘模块组成结构

Fig. 6　Structure of point multiplication module

点乘模块通过循环调用 256 次两个点运算模块完成 256 位的点乘计算，并采取移位寄存器搭配多路选择器的方式来判断是否需要进行点加。其中倍点与点加模块均采用了二级并行的方案与 Karatsuba-Ofman 算法模乘器进行加速。

3　SM4 算法的硬件加速设计

SM4 算法的高速实现是提升大量数据加解密速度的关键，在硬件实现上对 SM4 算法进行优化加速同样重要。本文对 S 盒的硬件实现进行了研究，采用复合域求逆的方式来构造；同时，还在此基础上提出一种二次流水全展开硬件架构，并对轮函数模块进行硬件复用，以较小硬件开销实现了极高的吞吐量。

3.1　基于复合域求逆 S 盒的实现

S 盒是 SM4 实现中对性能影响最大的部分，其在很大程度上决定了整个 SM4 算法的硬件实现性能。目前 S 盒的构造方法主要分为查表式与代数式。传统的查表式 S 盒是一个包含 256 个数据的字节代换表，虽然硬件上可以采用寄存器或 ROM 的方式简单实现，但是面积开销较大，电路性能也难以提升。代数式 S 盒又分为有限域求逆和复合域求逆两种构造方式。而有限域上的求逆运算非常复杂，难以用硬件实现；复合域求逆的方式则在运算量和硬件消耗上均很小，适合用以硬件实现。

在 2007 年 Liu 等[21]提出了一种用式(3)所示的代数式构造 S 盒的方法，即 S 盒可以通过仿射变换和乘法求逆来实现：

$$S(x)=I(x\cdot A_1+C_1)\cdot A_2+C_2 \tag{3}$$

其中，x 是 S 盒的 8 位二进制输入；A_1，A_2 为循环矩阵；C_1，C_2 为行向量；$I(x)$是 GF(28)域上的乘法求逆运算。在复合域上进行乘法求逆，可以在很大程度上简化运算复杂度。利用复合域求 GF(28)域上乘法逆元的运算过程如图 7 所示。

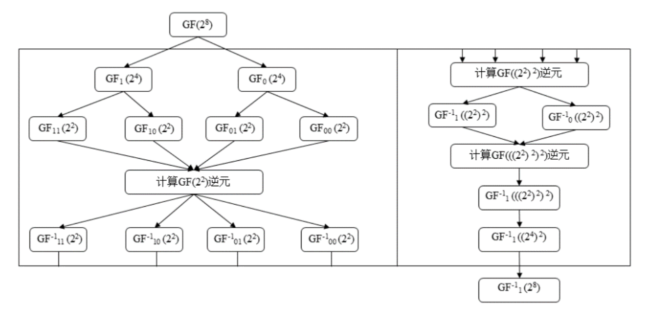


图 7　复合域乘法求逆运算过程

Fig. 7　The process of inverse transformation over composite field

其主要步骤如下：一、利用同构映射 T_1 和 T_2 将 GF(28)域上的元素映射到复合域 GF(((22)^2)^2)；二、在 GF(2)域上用异或运算求出 GF(22)的逆元；三、将 GF(22)的逆元结果回代运算，用基于 GF(22)的乘法运算求出 GF((22)^2)逆元；四、将 GF((22)^2)逆元结果回代运算，用基于 GF((22)^2)的乘法运算求出 GF(((22)^2)^2)的逆元；五、利用逆同构映射 T_1^{-1} 和 T_2^{-1} 将求逆结果从 GF(((22)^2)^2)映射回 GF(28)域，最终得到 GF(28)域的乘法逆元结果。

以⊕代表异或门，根据图 7 并参考文献[22]中关于复合域求逆的设计，对仿射变换和同构映射部分进行化简合并，设计出复合域求逆 S 盒的硬件结构如图 8~11 所示。

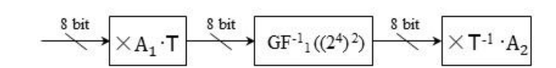


图 8　复合域 S 盒基本结构

Fig. 8　Structure of S-box based on the inverse transformation over composite field

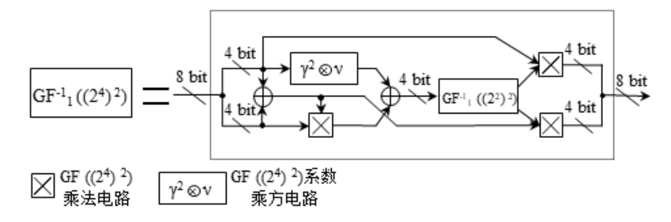


图 9　GF ((24) 2)逆元电路

Fig. 9　Hardware structure of inversion on GF ((24) 2)

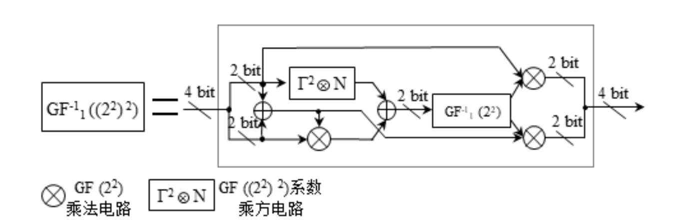


图 10　GF ((22) 2)逆元电路

Fig. 10　Hardware structure of inversion on GF ((22) 2)

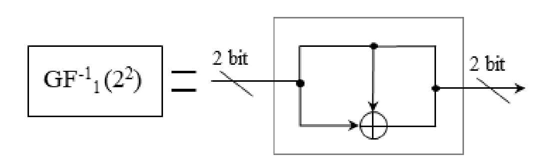


图 11 GF (22)逆元电路

Fig. 11 Hardware structure of inversion on GF (22)

可见，在最底层 GF(22)子域上的求逆运算仅仅只用到了异或运算。另外，根据扩展式对系数乘方运算进行化简优化后，设计出各域上的乘法电路与系数乘方电路硬件电路结构分别如图 12 和 13 所示。

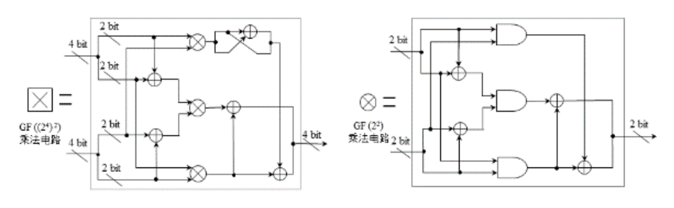


图 12 GF ((24) 2)和 GF (22)乘法电路

Fig. 12 Hardware structure of multiplication on GF ((24) 2) and GF (22)

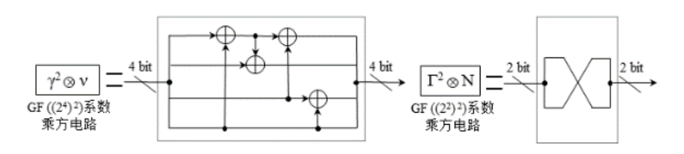


图 13 GF ((24) 2)和 GF ((22) 2)系数乘方电路

Fig. 13 Hardware structure of square on GF ((24) 2) and GF ((22) 2)

经优化后的 S 盒电路仅使用数量较少的异或门和与门即可实现 256 位数据的替换运算，相比于直接使用查找表的方式，该方案极大地降低了运算复杂度和硬件开销。同时， 采用纯组合逻辑实现的 S 盒电路也为后续进行轮内流水优化提供了便利。在基于复合域求逆 S 盒的基础上，根据 SM4 算法的描述，完成了 SM4 单轮轮函数的硬件设计，并分析各电路模块的资源消耗情况和关键路径如表 3 所示。

表 3 SM4 各电路模块的资源消耗与关键路径

Tab. 3 Gate counts and critical paths of functional blocks in SM4

电路模块	逻辑门数	关键路径
GF ((24) ^{2}) 系数乘方电路	4 XOR	2 XOR
GF (22)乘法电路	3 AND + 4 XOR	1 AND + 2 XOR
GF (24)乘法电路	6 AND + 13 XOR	1 AND + 4 XOR
GF (22)逆元电路	1 XOR	1 XOR
GF ((22) ^{2})逆元电路	9 AND + 15 XOR	2 AND + 7 XOR
GF ((24) ^{2})逆元电路	27 AND + 60 XOR	4 AND + 17 XOR
映射模块	30 XOR	7 XOR
逆映射模块	26 XOR	7 XOR
复合域 S 盒	27 AND + 116 XOR	4 AND + 31 XOR
SM4 单轮轮函数	108 AND + 472 XOR	4 AND + 36 XOR

3.2 SM4 算法的流水线结构设计

SM4 算法的吞吐率计算公式如式 4 所示，其中 B 代表处理的数据位宽，在这里固定为 128 位；f 代表电路频率；N 代表平均每组加解密数据的处理轮数。由此可见，要提高 SM4 算法的吞吐率一方面要对电路的关键路径进行优化，提升硬件整体频率 f，另一方面还要降低每组加解密数据的平均处理轮数 N。

$$Throughput = \frac{B \times f}{N} \tag{4}$$

根据 SM4 算法的定义可知，该算法是由 32 轮完全相同的轮函数组成，且在加解密过程中前后数据完全独立，因此可以采用流水线处理的结构来实现 SM4 分组密码算法的高吞吐率计算。本文首先采用轮内流水的结构实现单轮轮函数以提高时钟频率 f，再采用轮间流水的结构完成整体 SM4 算法电路以降低平均处理轮数 N，从而实现高速运算。

3.2.1 轮内流水处理结构

要想提高电路的时钟频率，就必须要降低每轮运算中的运算量，尤其是在轮函数运算较为复杂的时候，其关键路径限制了在硬件实现下的最大时钟频率。针对这种情况，可以采用轮内流水处理结构进行加速，通过将轮函数内的运算单元进行拆解后插入寄存器以建立流水线，进而构造轮内流水结构。

分析表 3 中 SM4 电路的关键路径，在轮函数电路中的不同位置插入寄存器，设计出如图 14 所示的四种轮内流水结构。轮内流水结构将每一轮的运算从内部切分，缩短了整体电路的关键路径，极大地降低了电路中每一拍的运算时间。

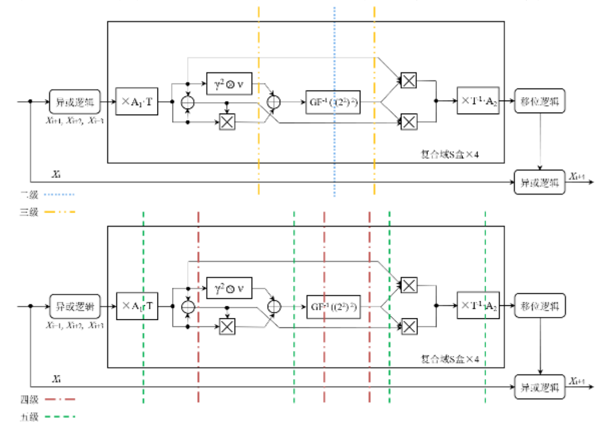


图 14 SM4 轮内流水结构

Fig. 14 Subpipelined structure of the round function circuit in SM4

3.2.2 轮间流水处理结构

一般而言，传统的 SM4 算法采取循环迭代式的结构实现，即仅实例化一个轮函数电路，并通过不断重复调用此电路以实现 32 轮的运算。对于传统循环迭代式结构，其每组数据的加解密时间至少为 32 个周期，下一组数据需要等待上一组数据循环计算 32 次后方可开始计算，实际吞吐率并不高。因此，该结构虽然能节省大量电路开销，但是其运算效率低下，不适合高性能加解密系统的设计。

而轮间流水处理结构则是指以轮函数作为基本运算单元，采取循环展开的方式，构造流水线，实现数据的流水处理。具体而言，轮间流水结构在进行电路设计时实例化 32 个轮函数电路，并在每一轮运算之间用寄存器隔开，每组数据无须等待，可以以流水的形式紧跟上组数据输入电路进行计算。轮间流水处理的优势在于，当流水线一旦构造完成，系统将在每个时钟周期完成一组密码运算，平均下来每组数据的加解密时间仅为一个周期，极大地提升了整体电路的吞吐率。

在采用图 14 所示的轮内流水结构的基础上，对 SM4 电路进行循环全展开，以轮间流水的方式实现 SM4 的整体电路设计，最终 SM4 算法的硬件电路架构如图 15 所示。

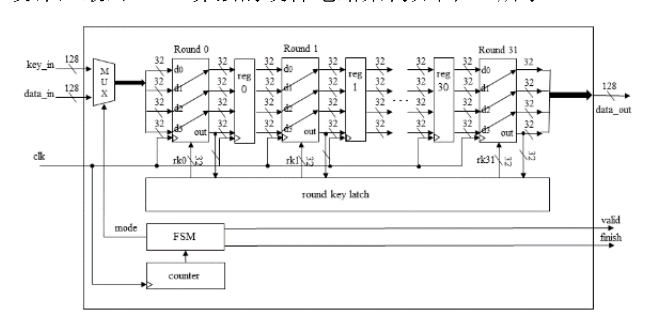


图 15 SM4 硬件电路架构

Fig. 15 High speed hardware architecture for SM4

该电路采用了复合域求逆 S 盒、轮内流水处理以及 32 轮全展开轮间流水处理三种优化技术，为加解密电路带来了极大的性能提升。同时，本文还根据加解密模块和轮密钥扩展模块在算法结构上的相似性，在轮函数电路内对这两个模块进行了硬件上的复用合并，并通过多路选择器来对输入数据以及移位逻辑进行选择，进一步地提高硬件电路的利用率。

4 实验结果与性能对比

4.1 实验环境

本文基于所述硬件架构，使用硬件描述语言 Verilog-HDL 实现了高速混合加密系统的整体硬件设计，并在硬件平台 Xilinx Ultrascale+系列的 ZCU102 开发板上进行板级测试，验证设计正确性及分析工作性能。

4.2 实验结果

本设计完成了一个高性能的混合加密系统，其系统框图如图 16 所示。该系统主要由 SM2、SM3、SM4 三种加密算法模块、状态机控制模块以及输入输出缓存器模块组成，可以采用三种加密算法进行高速防窜改混合加密运算。

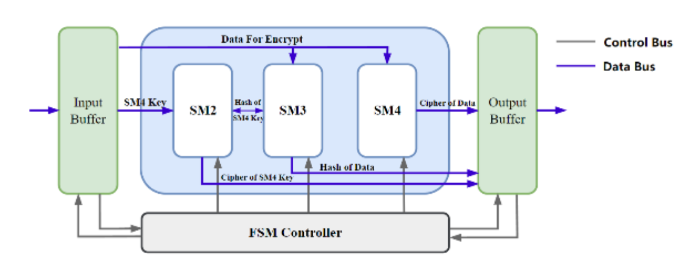


图 16 高速混合加密系统框图

Fig. 16 Structure of the high speed hybrid encryption system

在系统中三个密码模块相互独立且工作频率各不相同，模块间数据交互以异步方式处理。表 4 为总系统以及各算法模块工作频率与资源消耗情况。

表 4 混合加密系统及各算法模块硬件实现结果

Tab. 4 Hardware Implementaion results of the hybrid encryption system and each algorithm module

算法模块	Fmax(MHz)	资源消耗
SM2	215	41067 LUTs + 13837 Regs + 320 DSPs
SM3	200	1513 LUTs + 1744 Regs
SM4	400	14264 LUTs + 13874 Regs
混合加密系统	200	58313 LUTs + 36032 Regs + 320 DSPs

由表 4 中可看出，各算法模块的最高频率与资源消耗各不相同。其中 SM2 算法模块的面积开销最大，所需的 LUT 与 DSP 最多。而 SM3 模块由于采取了循环迭代的方式实现，极大地降低了电路资源消耗，电路开销最小。SM4 算法模块在考虑硬件效率的情况下选择了轮间全展开加上轮内四级流水的架构实现，其平均每组数据加密时间降低至一个时钟周期，同时工作频率也提升至 400MHz，非常适合大量数据的快速加解密运算。

此外，分别利用三种单一制算法加密大小为 16K 的明文数据，与本文的混合加密系统进行加密性能以及相关特性的对比，具体情况见表 5。

表 5 不同体制加密算法性能与特性对比

Tab. 5 Comparison of performance and characteristics of different encryption algorithms

加密算法	加密时间(μs)	双向加解密	额外安全信道
SM2	183.82	支持	不需要
SM3	24.70	不支持	不需要
SM4	0.64	支持	需要
混合加密	134.80	支持	不需要

可以看到，虽然 SM3 与 SM4 算法在加密时间上均有优势，但它们均有一些算法体制上的不足：SM3 算法只能进行单向加密计算，无法从杂凑值反推得到明文数据；SM4 算法需要额外的安全通道进行密钥的传输，在密钥管理上繁琐复杂。而混合加密算法不仅与 SM2 算法一样不存在上述算法体制的问题，而且在加密时间上也比 SM2 算法缩短了 26.67%，充分体现了本文设计的混合加密系统的优越性。

在 SM2 算法的点运算层优化上，本文的二级并行方案与传统串行方案[23]、三级并行方案[20]的复现结果性能对比如表 6、7 所示。其中 AT^2 为硬件电路的面积与运算时间平方的乘积，用以评估不同方案性能优劣。

表 6 点加方案的性能对比

Tab. 6 Comparison of performance for point addition schemes from different design

点加方案	资源消耗(LUT)	运算时间(μs)	AT2(×10²)
传统串行[23]	8988	0.32	9.204
二级并行(本文)	18141	0.175	5.556
三级并行[20]	24744	0.155	5.945

表 7 倍点方案的性能对比

Tab. 7 Comparison of performance for double point schemes from different design

倍点方案	资源消耗(LUT)	运算时间(μs)	AT2(×10²)
传统串行[23]	8382	0.235	4.629
二级并行(本文)	17045	0.155	4.095
三级并行[20]	24347	0.135	4.437

可以看到，本文设计的二级并行加速方案虽然在运算时间上并不是最短的，但是其 AT^2 指标均为最小，相比于传统串行方案分别降低了 39.6%和 11.5%，在运算时间和面积开销的协调平衡上最具优势。

在 SM4 算法的流水线优化上，表 8 则列出了在 32 轮全展开轮间流水的结构下不同轮内流水级数架构的 FPGA 电路实现情况。同时，表中还引入了吞吐率/面积的指标，以衡量各架构的速度与资源消耗情况。其中 CLB 代表 Xilinx FPGA 芯片的可配置逻辑单元(Configurable Logic Block, CLB)。

表 8 SM4 轮内流水结构

Tab. 8 Comparisons of FPGA implementations for SM4 different subpipelined structure

轮内流水级数	资源消耗				关键路径	Fmax	吞吐率	吞吐率/面积
	LUT	LUTRAM	Register	CLB		(MHz)	(Mbps)	(Mbps/CLB)
No	12372	NA	6419	1547	4 AND + 36 XOR	169	21632	13.98
2	14852	NA	12214	1857	2 AND + 18 XOR	259	33152	17.85
3	12161	496	14912	1963	1 AND + 14 XOR	334	42752	21.77
4	14264	1488	13874	2110	2 AND + 8 XOR	400	51200	24.26
5	13216	1504	14624	2239	2 AND + 7 XOR	420	53760	24.01

由表 8 可以看到，五级轮内流水架构的时钟频率和吞吐率均为最高，分别达到了 420MHz 和 53.7Gbps，吞吐率相比无轮内流水提高了 136.7%。而在吞吐率/面积指标上，四级轮内流水架构取得了最佳的性能，高达 24.26 Mbps/CLB，相比优化前提升了 73.5%。在对加密速率和整体硬件消耗进行取舍后，选取四级轮内流水架构作为混合加密系统中 SM4 算法的实现方案，以实现高吞吐率与硬件消耗的性能平衡。

4.3　相关工作对比

4.3.1 SM2 性能对比

将 SM2 算法模块的硬件实现结果与其他同类型文献工作对比，得出如表 9 所示的 SM2 算法点乘运算性能对比情况。

表 9　点乘运算硬件实现性能对比

Tab. 9　Comparison of performance from different hardware implementation of point multiplication

文献	平台	资源消耗	Fmax/MHz	运算时间/μs
文献[23]	40nm	127K Gates	388	480
文献[18]	130nm	659K Gates	163.7	20.36
文献[24]	130nm	352.5K Gates	100	305.82
文献[25]	130nm	248.9K Gates	200	71.5
文献[7]	Stratix II	4742ALMs+8DSPs	62.3	770
文献[20]	Virtex-7	29876LUTs+144DSPs	73.6	221
文献[27]	Virtex-2	12425LUTs	55.7	8250
文献[28]	Virtex-2	36524LUTs	63.2	1470
本文	**ZCU102**	**41067LUTs+320DSPs**	**215**	**68.37**

从表 9 可知，本文设计的 SM2 点乘电路的最高工作频率高达 215MHz，仅需 68.37μs 即可完成一次点乘计算，在运算速度上具有一定优势。经本文优化后的点乘电路仅次于文献[18]，而文献[18]由于采用了多个 256 位大位宽乘法器，可以在一个周期内完成模乘计算，虽然具有高性能的优势，但是硬件开销也非常庞大，并不适合资源有限的场景。此外，在同样使用 FPGA 开发平台的工作中，本设计在工作频率和运算速度上的优势更为明显：与综合性能较优的文献[20]相比，本设计的最高频率提升了约 2.92 倍，运算时间缩短了约 68.9%。

综上所述，本设计的模乘单元采用了 KO 算法和模加运算并行的实现方案，有效地降低了模乘运算的运算周期，同时还对点运算层进行二级模乘运算并行的优化设计，使得优化后的 SM2 算法电路取得最佳的性能，更加适合本文混合加密系统。

4.3.2 SM4 性能对比

表 10 中列举了同样在 32 轮全展开轮间流水的结构下，本文的 SM4 设计方案与各文献方案的电路实现情况性能对比，表中的吞吐率均为理论峰值吞吐率。相比于各参考文献，在同等采用轮间流水优化结构的情况下，本文设计的四级与五级轮内流水架构突破了单轮运算关键路径过长的瓶颈，其系统最高工作频率分别可达到 400MHz 和 420MHz，峰值吞吐率分别高达 51.2Gbit/s 和 53.76Gbit/s。与性能较优的文献[30]相比，本设计的最高频率分别提升了 16.3%和 22%，吞吐率提高了 21.6%和 27.7%，完成了高吞吐率加解密算法硬件实现的目标，可以很好地满足本文提出的高速混合加密系统的需求。

5　结束语

本文提出了一种基于国密算法 SM2、SM3、SM4 的高速混合加密系统，通过混合使用三种加密算法避免了单一密码体制安全性不高、加密速率过慢、密钥管理不便等缺点。此外，本文还对系统中限制整体加解密速度提升的 SM2 和 SM4 算法进行硬件实现上的优化设计。实验结果表明，本文设计的 SM2 硬件电路由于采用了二级点运算并行架构，在运算速率与硬件利用率上相比于现有工作更具优势。另外，本文在轮间流水结构的基础上，对内部电路进行分析与优化，设计出了一种基于复合域 S 盒的 SM4 二次流水架构，进一步提高了电路工作频率，其最高峰值吞吐率达到了 53.76Gbit/s。综上所述，本文所设计的混合加密系统通过采用多种加密算法以及对相关电路进行优化，在保证安全性的同时提供了极高的运算速率，解决了大数据时代下海量数据加解密的高速加解密需求。

表 10　SM4 硬件设计性能对比

Tab. 10　Comparison of performance from different hardware implementation of SM4

文献	平台	资源消耗	Fmax/MHz	吞吐率/Gbps
文献[10]	Stratix II	8373 ALMs	162	20.736
文献[11]	Stratix II	7661 ALMs	170	21.760
文献[26]	Stratix IV	8045 ALMs	212.13	27.153
文献[13]	Virtex-4	7669 CLBs	298	26.224
文献[29]	Cyclone IV	33397 LEs	100	12.8
文献[30]	Kintex-7	3213 LUTs +7736 Regs +33BRAMs	344	42.1
本文(四级轮内流水)	ZCU102	2110 CLBs	400	51.200
本文(五级轮内流水)	ZCU102	2239 CLBs	420	53.760

参考文献：

[1] Mostafaa H, Eisaa S M, Issaa H H, *et al.* Lightweight hybrid encryption system with FPGA design proposal [J]. IOP Conference Series: Materials Science and Engineering, 2021, 1051 (1): 012023.

[2] Shende V, Kulkarni M. FPGA based hardware implementation of hybrid cryptographic algorithm for encryption and decryption [C]// Proc of IEEE ICEECCOT. Piscataway, NJ: IEEE Press, 2017: 416-419.

[3] 卞建秀，李垣江，王建华. 基于 SM4 和 ECC 的混合加密算法研究 [J]. 计算机应用与软件, 2016, 33 (10): 303-306, 324. (Bian Jianxiu, Li Yuanjiang, Wang Jianhua. Study on SM4 and ECC-based hybrid encryption algorithm [J]. Computer Applications and Software, 2016, 33 (10): 303-306, 324.)

[4] 方轶，丛林虎，邓建球. 基于国密算法的武器装备数据混合加密方案 [J]. 探测与控制学报, 2020, 42 (01): 121-126. (Fang Yi, Cong Linhu, Deng Jianqiu. A hybrid encryption scheme of weapons and equipment data based on national security algorithm [J]. Journal of Detection and Control, 2020, 42 (01): 121-126.)

[5] Wang Tong, Cui Wenpeng, Li Tong, *et al.* The research of the SM2, SM3 and SM4 algorithms in WLAN of transformer substation [C]// Proc of the 3rd International Conference on Electronic Information Technology and Computer Engineering. Piscataway, NJ: IEEE Press, 2019: 276-283.

[6] Zheng Xin, Xu Chongyao, Hu Xianghong, *et al.* The software/hardware co-design and implementation of SM2/3/4 encryption/decryption and digital signature system [J]. IEEE Trans on Computer-Aided Design of Integrated Circuits and Systems, 2019, 39 (10): 2055-2066.

[7] 郭晓，蒋安平，宗宇. SM2 高速双域 Montgomery 模乘的硬件设计 [J]. 微电子学与计算机, 2013, 30 (9): 5. (Guo Xiao, Jiang Anping, Zong Yu. A high speed structure for dual-field montgomery modular multiplication in SM2 [J]. Microelectronics and Computer, 2013, 30 (9): 5.)

[8] Zhang Dan, Bai Guoqiang. High-performance implementation of SM2 based on FPGA [C]// Proc of the 8th IEEE International Conference on Communication Software and Networks. Piscataway, NJ: IEEE Press, 2016: 718-722.

[9] 杨博，孟李林，陶琼. 基于 FPGA 的 F_P 域模乘与模逆的设计与实现 [J]. 微电子学与计算机, 2017, 34 (5): 5. (Yang Bo, Meng Lilin, Tao

Qiong. Design of F_P field modular multiplication and modular inversion based on FPGA [J]. Microelectronics and Computer, 2017, 34 (5): 5.)

[10] Gao Xianwei, Lu Erhong, Xian Liqin, *et al.* FPGA implementation of the SMS4 block cipher in the Chinese WAPI standard [C]// Proc of IEEE ICESS. Piscataway, NJ: IEEE Press, 2008: 104-106.

[11] Wang Husen, Li Shuguo. High performance FPGA implementation for SMS4 [M]. Berlin, Heidelberg: Springer, 2011: 469-475.

[12] 王晨光，乔树山，黑勇．分组密码算法 SM4 的低复杂度实现 [J]. 计算机工程, 2013, 39 (7): 4. (Wang Chenguang, Qiao Shushan, Hei Yong. Low complexity implementation of block cipher SM4 algorithm [J]. Computer Engineering, 2013, 39 (7): 4.)

[13] 周洲，何一凡，沈海斌，等．SMS4 密码算法高速引擎实现 [J]. 电子器件, 2007 (04): 347-349, 358. (Zhou Zhou, He Yifan, Shen Haibin, *et al.* High-speed implementation of the SMS4 algorithm engine [J]. Journal of Electron Devices, 2007 (04): 347-349, 358.)

[14] Fu Hailiang, Bai Guoqiang, Wu Xingjun. A very compact masked S-Box for high-performance implementation of SM4 based on composite field [C]// Proc of ICST. Berlin, Heidelberg: Springer, 2017: 710-721.

[15] 汪朝晖，张振峰. SM2 椭圆曲线公钥密码算法综述 [J]. 信息安全研究, 2016, 2 (11): 972-982. (Wang Zhaohui, Zhang Zhenfeng. Overview on public key cryptographic algorithm SM2 based on elliptic curves [J]. Journal of Information Security Research, 2016, 2 (11): 972-982.)

[16] 吕述望，苏波展，王鹏，等．SM4 分组密码算法综述 [J]. 信息安全研究, 2016, 2 (11): 13. (Lyu Shuwang, Su Bozhan, Wang Peng, *et al.* Overview on SM4 algorithm [J]. Journal of Information Security Research, 2016, 2 (11): 13.)

[17] 邹雪城，周家乐，刘文超，等．小面积高兼容性 RSA&SM2 的硬件实现方法 [J]. 华中科技大学学报：自然科学版, 2019, 47 (01): 79-84. (Zou Xuecheng, Zhou Jiale, Liu Wenchao, *et al.* Design method of RSA&SM2 hardware with low-area and high-compatibility [J]. Journal of Huazhong University of Science and Technology: Natural Science, 2019, 47 (01): 79-84.)

[18] Zhao Zhenwei, Bai Guoqiang. Ultra high-speed SM2 ASIC implementation [C]// Proc of the 13th IEEE International Conference on Trust, Security and Privacy in Computing and Communications. Piscataway, NJ: IEEE Press, 2014: 182-188.

[19] Hankerson D, Menezes A J, Vanstone S. Guide to elliptic curve cryptography [M]. [S. l.] : Springer Science & Business Media, 2006.

[20] 李凡，李云峰，翁天恒，等．基于 FPGA 的 SM2 点运算快速并行实现 [J]. 电子测量技术, 2020 (15): 7. (Li Fan, Li Yunfeng, Weng Tianheng, *et al.* Implementation of parallel and fast SM2 point calculation on FPGA [J]. Electronic Measurement Technology, 15 (2020): 7.)

[21] Liu Fen, Wen Ji, Lei Hu, *et al.* Analysis of the SMS4 block cipher [C]// Proc of ACISP. Berlin, Heidelberg: Springer, 2007: 158-170.

[22] 徐艳华，白雪飞，郭立．适合 SMS4 算法硬件实现的 S 盒构造新方法 [J]. 中国科学技术大学学报, 2009 (11): 7. (Xu Yanhua, Bai Xuefei, Guo Li. A new algorithm of S-box for hardware implementation of SMS4 [J]. Journal of University of Science and Technology of China, 2009 (11): 7.)

[23] 陆启乐．一种面向椭圆曲线的双域点乘加速器的设计 [D]. 南京：东南大学, 2018. (Lu Qile. Design of a dual-field point multiplication accelerator in elliptic curve cryptography [D]. Nan Jing: Southeast University, 2018.)

[24] 陆江城．基于非对称加密算法的加密系统的研究与实现 [D]. 广州：广东工业大学, 2020. (Lu Jiangcheng. Research and implementation of encryption system based on asymmetric encryption [D]. Guangzhou: Guangdong University of Technology, 2020.)

[25] 张盛仕，胡湘宏，熊晓明．基于国密算法 SM2 软硬件协同系统的 FPGA 架构 [J]. 单片机与嵌入式系统应用, 2019, 19 (7): 5. (Zhang Shengshi, Hu Xianghong, Xiong Xiaoming. FPGA architecture of software and hardware co-design based on nation secret algorithm SM2 [J]. Microcontrollers and Embedded Systems, 2019, 19 (7): 5.)

[26] Guan Zhenyu, Li Yunhao, Shang Tao, *et al.* Implementation of SM4 on FPGA: trade-off analysis between area and speed [C]// Proc of IEEE ISR. Piscataway, NJ: IEEE Press, 2018: 192-197.

[27] Liu Z, Liu D, Zou X. An efficient and flexible hardware implementation of the dual-field elliptic curve cryptographic processor [J]. IEEE Trans on Industrial Electronics, 2017, 64 (3): 2353-2362.

[28] 张丽．双域椭圆曲线密码协处理器关键技术研究 [D]. 西安：西安电子科技大学, 2019. (Zhang Li. Research on key technology of dual-field elliptic curve cryptography coprocessor [D]. Xian: Xidian University, 2019.)

[29] 刘金峒，梁科，王锦，等．SM4 加密算法可裁剪式结构设计与硬件实现 [J]. 南开大学学报：自然科学版, 2019, 52 (4): 5. (Liu Jintong, Liang Ke, Yu Jin, *et al.* Tailorable structure design and hardware implementation of SM4 encryption algorithm [J]. Journal of Nankai University: Natural Science, 2019, 52 (4): 5.)

[30] 何诗洋，李晖，李凤华．SM4 算法的 FPGA 优化实现方法 [J]. 西安电子科技大学学报, 2021, 48 (3): 8. (He Shiyang, Li Hui, Li Fenghua. Optimization and implementation of the SM4 on FPGA [J]. Journal of Xidian University, 2021, 48 (3): 8.)